## CRC calculation application note

## 8-bit CRC calculation with 0x97 polynome

Some of the communication interfaces offer a CRC value to check the correctness of the data read from the encoder. This chapter gives an example of the CRC calculation on the receiver side. The CRC calculation must always be done over the complete set of data. The polynomial for the CRC calculation is $\mathrm{P}(\mathrm{x})=\mathrm{x} 8+\mathrm{x} 7+\mathrm{x} 4+\mathrm{x} 2+\mathrm{x} 1+1$, also represented as $0 \times 97$.

## Code example:

//Lookup table for polynome $=0 \times 97$
static const u8 ab_CRC8_LUT[256] = \{
$0 \times 00,0 \times 97,0 \times B 9,0 \times 2 \bar{E}, 0 \times E 5,0 \times 72,0 \times 5 \mathrm{C}, 0 \times \mathrm{CB}, 0 \times 5 \mathrm{D}, 0 \times \mathrm{CA}, 0 \times \mathrm{E} 4,0 \times 73,0 \times B 8,0 \times 2 \mathrm{~F}, 0 \times 01,0 \times 96$, $0 x B A, 0 x 2 D, 0 x 03,0 x 94,0 x 5 F, 0 x C 8,0 x E 6,0 x 71,0 x E 7,0 x 70,0 \times 5 E, 0 x C 9,0 \times 02,0 x 95,0 \times B B, 0 x 2 C$, $0 x E 3,0 \times 74,0 x 5 \mathrm{~A}, 0 \times \mathrm{CD}, 0 \times 06,0 \times 91,0 \times B E, 0 \times 28,0 x B E, 0 \times 29,0 \times 07,0 \times 90,0 \times 5 \mathrm{~B}, 0 \times C C, 0 \times \mathrm{E} 2,0 \times 75$, $0 x 59,0 x C E, 0 x E 0,0 x 77,0 x B C, 0 x 2 B, 0 x 05,0 x 92,0 x 04,0 x 93,0 x B D, 0 x 2 A, 0 x E 1,0 x 76,0 x 58,0 x C F$, $0 \times 51,0 \times C 6,0 \times E 8,0 \times 7 \mathrm{~F}, 0 \times \mathrm{B} 4,0 \times 23,0 \times 0 \mathrm{D}, 0 \times 9 \mathrm{~A}, 0 \times 0 \mathrm{C}, 0 \times 9 \mathrm{~B}, 0 \times \mathrm{B} 5,0 \times 22,0 \times \mathrm{P} 9,0 \times 7 \mathrm{E}, 0 \times 50,0 \times C 7$, $0 x E B, 0 \times 7 \mathrm{C}, ~ 0 \times 52,0 \times C 5,0 \times 0 \mathrm{E}, 0 \times 99,0 \times B 7,0 \times 20,0 \times B 6,0 \times 21,0 \times 0 \mathrm{~F}, 0 \times 98,0 \times 53,0 \times C 4,0 \times E A, 0 x 7 \mathrm{D}$, $0 \times B 2,0 \times 25,0 \times 0 B, 0 \times 9 C, 0 \times 57,0 \times C 0,0 \times E E, 0 \times 79,0 x E F, 0 \times 78,0 \times 56,0 \times C 1,0 \times 0 A, 0 x 9 D, 0 \times B 3,0 \times 24$, $0 \times 08,0 \times 9 \mathrm{~F}, 0 \times B 1,0 \times 26,0 \times E D, 0 x 7 A, 0 \times 54,0 \times C 3,0 \times 55,0 \times C 2,0 x E C, 0 \times 7 B, 0 x B 0,0 \times 27,0 \times 09,0 x 9 E$, $0 \times A 2,0 \times 35,0 \times 1 \mathrm{~B}, 0 \times 8 \mathrm{C}, 0 \times 47,0 \times \mathrm{D} 0,0 \times F E, 0 \times 69,0 \times F F, 0 \times 68,0 \times 46,0 \times D 1,0 \times 1 \mathrm{~A}, 0 \times 8 \mathrm{D}, 0 \times \mathrm{A} 3,0 \times 34$, $0 x 18,0 x 8 F, 0 x A 1,0 x 36,0 x F D, 0 x 6 A, 0 x 44,0 x D 3,0 x 45,0 x D 2,0 x F C, 0 x 6 B, 0 x A 0,0 x 37,0 x 19,0 x 8 E$, $0 \times 41,0 x D 6,0 \times F 8,0 \times 6 \mathrm{~F}, 0 \times \mathrm{A} 4,0 \times 33,0 \times 1 \mathrm{D}, 0 \times 8 \mathrm{~A}, 0 \times 1 \mathrm{C}, 0 \times 8 \mathrm{~B}, 0 \times \mathrm{A} 5,0 \times 32,0 \times F 9,0 \times 6 \mathrm{E}, 0 \times 40,0 \times D 7$, $0 \times F B, 0 \times 6 \mathrm{C}, 0 \times 42,0 \times D 5,0 \times 1 \mathrm{E}, 0 \times 89,0 \times A 7,0 \times 30,0 \times A 6,0 \times 31,0 \times 1 \mathrm{~F}, 0 \times 88,0 \times 43,0 \times D 4,0 \times F A, 0 \times 6 \mathrm{D}$, $0 x F 3,0 x 64,0 x 4 A, 0 x D D, 0 x 16,0 x 81,0 x A F, 0 x 38,0 x A E, 0 x 39,0 x 17,0 x 80,0 x 4 B, 0 x D C, 0 x F 2,0 x 65$, $0 \times 49,0 \times D E, 0 \times F 0,0 x 67,0 \times A C, 0 x 3 B, 0 \times 15,0 \times 82,0 \times 14,0 \times 83,0 \times A D, 0 \times 3 A, 0 \times F 1,0 \times 66,0 \times 48,0 \times D F$, $0 \times 10,0 \times 87,0 x A 9,0 x 3 E, 0 \times F 5,0 x 62,0 x 4 \mathrm{C}, 0 \times D B, 0 x 4 \mathrm{D}, 0 \times D A, 0 x F 4,0 \times 63,0 \times A 8,0 \times 3 F, 0 \times 11,0 \times 86$, $0 x A A, 0 x 3 D, 0 x 13,0 x 84,0 x 4 F, 0 x D 8,0 x F 6,0 x 61,0 x F 7,0 x 60,0 x 4 E, 0 x D 9,0 x 12,0 x 85,0 x A B, 0 x 3 C\}$;
/* CRC 0x97 Polynomial, 64-bit input data, right alignment, calculation over 64 bits */

```
u8 CRC_SPI_97_64bit(u64 dw_InputData)
{
    u8 b_Index = 0;
    u8 b_CRC = 0;
    b_Index = (u8) ((dw_InputData >> 56u) & (u64)0x000000FFu);
    b_CRC = (u8) ((dw_InputData >> 48u) & (u64)0x000000FFu);
    b Index = b CRC ^ ab CRC8 LUT[b Index];
    b CRC = (u8) ((dw InputData >> 40u) & (u64)0x000000FFu);
    b_
    b_CRC = (u8) ((dw_InputData >> 32u) & (u64)0x000000FFu);
    b_Index = b_CRC ^ ab_CRC8_LUT[b_Index];
    b_CRC = (u8) ((dw_InputData >> 24u) & (u64)0x000000FFu);
    b_Index = b_CRC ^ ab_CRC8_LUT[b_Index];
    b_CRC = (u8) ((dw_InputData}>>>1\overline{6}u) & (u64)0x000000FFu)
    b_Index = b_CRC ^ ab_CRC8_LUT[b_Index];
    b_CRC = (u8) ((dw_InputData >> 8u) & (u64)0x000000FFu);
    b_Index = b_CRC ^}\mp@subsup{}{}{\wedge}\textrm{ab}_CRC8_LUT [b_Index]
    b CRC = (u8) (dw InputData & (u64)0x000000FFu);
    b_Index = b_CRC ^ ab_CRC8_LUT[b_Index];
    b_CRC = ab_C
    return b_CRC;
}
```


## Example:

```
uint8_t rx_buffer[numOfBytes]; // contains received bytes
// TODO: load rx_buffer array with received data from the encoder
uint64_t dw_CRCinputData = 0;
uint8_t calculated_crc=0;
dw_CRCinputData = ((uint64_t)rx_buffer[0] << 32) + ((uint64_t)rx_buffer[1] << 24) +
    ((uint64_t) rx_buffer[2] << 16) + ((uint64_t) rx_buffer[3] << 8) +
    ((uint64 t) rx buffer[4] << 0);
calculated_crc = ~ (CRC_SPI_97_64bit(dw_CRCinputData))& 0xFF; //inverted CRC
```


## CRCD01_03

## 6-bit CRC calculation with $0 \times 43$ polynome for BiSS

BiSS communication offers a CRC value to check the correctness of the data read from the encoder. This chapter gives an example of the CRC calculation on the receiver side. The CRC calculation must always be done over the complete set of data. The polynomial for the CRC calculation is $P(x)=x 6+x 1+1$, also represented as $0 \times 43$.
Following code example must be modified to fit actual data length. Position data, error and warning bits must be included into calculation in the same order as in the BiSS data packet. ACK, Start and CDS bits are not included in the CRC calculation.

## Code example:

```
uint8_t tableCRC6[64] = {
    0x00, 0x03, 0x06, 0x05, 0x0C, 0x0F, 0x0A, 0x09,
    0x18, 0x1B, 0x1E, 0x1D, 0x14, 0x17, 0x12, 0x11,
    0x30, 0x33, 0\times36, 0\times35, 0x3C, 0x3F, 0x3A, 0x39,
    0x28, 0x2B, 0x2E, 0x2D, 0x24, 0x27, 0x22, 0x21,
    0x23, 0x20, 0x25, 0x26, 0x2F, 0x2C, 0x29, 0x2A,
    0x3B, 0x38, 0x3D, 0x3E, 0\times37, 0\times34, 0\times31, 0\times32,
    0x13, 0x10, 0\times15, 0\times16, 0x1F, 0x1C, 0x19, 0x1A,
    0x0B, 0x08, 0x0D, 0x0E, 0x07, 0x04, 0x01, 0x02};
/*32-bit input data, right alignment, Calculation over 24 bits (mult. of 6) */
uint8_t CRC_BiSS_43_24bit (uint32_t w_InputData)
{
    uint8_t b_Index = 0;
    uint8_t b_CRC = 0;
    b_Index = (uint8_t )(((uint32_t)w_InputData >> 18u) & 0x0000003Fu);
    b_CRC = (uint8_t )(((uint32_t)w_InputData >> 12u) & 0x0000003Fu);
    b_Index = b_CRC ^ ab_CRC6_LUT[b_Index];
    b_CRC = (uint8_t )(((uint32_t)w_InputData >> 6u) & 0x0000003Fu);
    b_
    b_CRC = (uint8_t ) ((uint32_t)w_InputData & 0x0000003Fu);
    b_Index = b_CR\overline{C}}^\mathrm{ ^ ab_CRC6_LUT [b_Index];
    b_CRC = ab_CRC6_LUT [b_Index];
    return b_CRC;
}
/*64-bit input data, right alignment, Calculation over 42 bits (mult. of 6) */
uint8_t CRC_BiSS_43_42bit(uint64_t dw_InputData)
{
    uint8_t b_Index = 0;
    uint8_t b_CRC = 0;
    b_Index = (uint8_t)((dw_InputData >> 36u) & (uint64_t)0x00000003Fu);
    b_CRC = (uint8_t) ((dw_InputData >> 30u) & (uint64_t)0x0000003Fu);
    b_Index = b_CRC ^ ab_CRC6_LUT[b_Index];
    b_CRC = (uint8_t) ((dw_InputData >> 24u) & (uint64_t)0x0000003Fu);
    b_Index = b_CRC ^ ab_CRC6_LUT[b_Index];
    b_CRC = (uint8_t) ((dw_InputData >> 18u) & (uint64_t)0x0000003Fu);
    b_Index = b_CRC ^ ab_CRC6_LUT[b_Index];
    b_CRC = (uint8_t)((dw_InputData >> 12u) & (uint64_t)0x0000003Fu);
    b_Index = b_CR\overline{C}}^\mathrm{ ^ ab_प्CRC6_LUT[b_Index];
    b_CRC = (uint8_t)((dw_InputData >> 6u) & (uint64_t)0x0000003Fu);
    b_Index = b_CR\overline{C}}^\mathrm{ ^ ab_㱜C6_LUT[b_Index];
    b_CRC = (uint8_t) (dw_InputData & (uint64_t) 0x0000003Fu);
    b_Index = b_CRC ^ ab_CRC6_LUT[b_Index];
    b_CRC = ab_CRC6_LUT[b_Index];
    return b_CRC;
}
```


## Recommended literature:

Painless guide to CRC error detection algorithm; Ross N. Williams.

- Cyclic Redundancy Code (CRC) Polynomial Selection For Embedded Networks; P. Koopman, T. Chakravarty


## Head office

## RLS merilna tehnika d.o.o.

| Poslovna cona Žeje pri Komendi | T +38615272100 |
| :--- | :--- |
| Pod vrbami 2 | F +38615272129 |
| SI-1218 Komenda | E mail@rls.si |
| Slovenia |  |

www.rls.si

## Global support

Visit our website to contact your nearest sales representative.

This product is not designed or intended for use outside the environmental limitations and operating parameters expressly stated on the product's datasheet. Products are not designed or intended for use in medical, military, aerospace, automotive or oil \& gas applications or any safety-critical applications where a failure of the product could cause severe environmental or property damage, personal injury or death. Any use in such applications must be specifically agreed to by seller in writing, and is subject to such additional terms as the seller may impose in its sole discretion. Use of products in such applications is at buyer's own risk, and buyer will indemnify and hold harmless seller and its affiliates against any liability, loss, damage or expense arising from such use. Information contained in this datasheet was derived from product testing under controlled laboratory conditions and data reported thereon is subject to the stated tolerances and variations, or if none are stated, then to tolerances and variations consistent with usual trade practices and testing methods. The product's performance outside of laboratory conditions, including when one or more operating parameters is at its maximum range, may not conform to the product's datasheet. Further, information in the product's datasheet does not reflect the performance of the product in any application, end-use or operating environment buyer or its customer may put the product to. Seller and its affiliates make no recommendation, warranty or representation as to the suitability of the product for buyer's application, use, end-product, process or combination with any other product or as to any results buyer or its customer might obtain in their use of the product. Buyer should use its own knowledge, judgment, expertise and testing in selecting the product for buyer's application, end-use and/or operating environment, and should not rely on any oral or written statement, representation, or samples made by seller or its affiliates for any purpose. EXCEPT FOR THE WARRANTIES EXPRESSLY SET FORTH IN THE SELLER'S TERMS AND CONDITIONS OF SALE, SELLER MAKES NO WARRANTY EXPRESS OR IMPLIED WITH RESPECT TO THE PRODUCT, INCLUDING ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE, WHICH ARE DISCLAIMED AND EXCLUDED. All sales are subject to seller's exclusive terms and conditions of sale which, where the seller is (a) RLS Merilna tehnika d.o.o., are available at https://www.rls.si/eng/salesterms, (b) Renishaw, Inc., are available at https:// www.renishaw.com/legal/en/--42186, or (c) another person, are available on request, and in each case, are incorporated herein by reference, and are the exclusive terms of sale. No other terms and conditions apply. Buyer is not authorized to make any statements or representations that expand upon or extend the environmental limitations and operating parameters of the products, or which imply permitted usage outside of that expressly stated on the datasheet or agreed to in writing by seller.

RLS Merilna tehnika d.o.o. has made considerable effort to ensure the content of this document is correct at the date of publication but makes no warranties or representations regarding the content. RLS Merilna tehnika d.o.o. excludes liability, howsoever arising, for any inaccuracies in this document. © 2020 RLS d.o.0.

